IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

APPLICANT:	Jean Khawand	ART UNIT:	2182
APPLN. NO.:	10/643,327	EXAMINER:	Sorrell, Eron J
FILED:	August 19, 2003		
TITLE:	METHOD AND APPARATUS FOR PROVIDING INTERPROCESSOR COMMUNICATIONS USING SHARED MEMORY		
Confirmation No.	4001		

CERTIFICATE UNDER 37 CFR 1.8(a)	
I hereby certify that this correspondence is being electronically transmitted on the date listed below:	
Date:	January 23, 2008
Signature	/Larry G. Brown/
Typed or printed name:	Larry G. Brown

APPEAL BRIEF UNDER 37 C.F.R. 41.37

Mail Stop: **APPEAL BRIEF-PATENTS**
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Attention: Board of Patent Appeals and Interferences

Dear Chief Administrative Patent Judge:

This Appeal Brief is in response to the Notice of Panel Decision from Pre-Appeal Brief Review of November 29, 2007 and is in furtherance of Applicants' Notice of Appeal filed on September 13, 2007.

Concurrently with this submission, applicants are paying the requisite fee for a one-month Extension of Time, and the requisite fee for filing this Appeal Brief.

I. REAL PARTY IN INTEREST

The real party of interest is Motorola, Inc., a Delaware corporation.

II. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

III. STATUS OF CLAIMS

This is an appeal from the final rejection of claims 1-19 of the above-referenced application.

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

There are a total of 19 claims in the application.

B. STATUS OF ALL THE CLAIMS

1. Claims allowed: none
2. Claims objected to: none
3. Claims rejected: 1-19
4. Claims cancelled: none

C. CLAIMS ON APPEAL

The claims on appeal are 1-19.

IV. STATUS OF AMENDMENTS

No amendments have been filed subsequent to the Final Rejection.

V. SUMMARY OF CLAIMED SUBJECT MATTER

Although specification citations are inserted below in accordance with C.F.R. 41.37, these reference numerals and citations are merely examples of where support may be found in the specification for the terms used in this section of the brief. There is no intention to in any way suggest that the terms of the claims are limited to the examples in the specification. Although, as demonstrated by the reference numerals and citations below, the claims are fully supported by the specification as required by law, it is improper under the law to read limitations from the specification into the claims. Pointing out specification support for the claim terminology, as is done here to comply with rule 41.37, does not in any way limit the scope of the claims to those examples from which they find support. Nor does this exercise provide a mechanism for circumventing the law precluding reading limitations into the claims from the specification. In short, the reference numerals and specification citations are not to be construed as claim limitations or in any way used to limit the scope of the claims.

The claimed subject matter of independent claim 1 pertains to an electronic device (200) that includes a first processor (102, 202), a second processor (104, 204) coupled to the first processor (102, 202) and a shared memory (112, 206) coupled to the first and second processors (102, 202; 104, 204) (see FIGs. 1 and 2 and page 3, lines 5-9 and page 4, lines 12-15 of the application). In addition, the shared memory (112, 206) includes the transmit memories of both the first and second processors (102, 202; 104, 204) (see page 3, lines 3-5 of the application).

The first processor (102, 202) is a master processor that manages the shared memory (112, 206) (see page 3, lines 3-9 of the application) and allocates a message buffer (114) to the second processor (104, 204) based on a specific request from the second processor (104, 204) to send a message to the first processor (102, 202) (see page 3, lines 11-20 of the application). Also, the first processor (102, 202) sends a message buffer pointer to the second processor (104, 204) that directs the second processor (104, 204) to the message buffer (114) (see page 3, lines 14-17 of the application).

The claimed subject matter of independent claim 11 pertains to a method for providing interprocessor communication between first and second processors (102, 202; 104, 204) using a shared memory (112, 206) that includes the transmit memories of the first and second processors (102, 202; 104, 204) (see FIGs. 1 and 2 and page 3, lines 3-9 and page 4, lines 12-15 of the application). The first processor (102, 202) is a master processor that is assigned to manage the shared memory (112, 206) (see page 3, lines 3-9 of the application).

The method includes the steps of sending a request from the second processor (104, 204) requesting an empty message buffer (114) from the shared memory (112, 206) when the second processor (104, 204) needs to send a message to the first processor (102, 202) and sending a message buffer pointer from the first processor (102, 202) to the second processor (104, 204) in response to this request (see FIG. 4 (steps 402 and 404) and page 5, line 20 to page 6, line 1 of the application). The method also includes the steps of using the message

buffer pointer by the second processor (104, 204) to locate the empty message buffer (114) in the shared memory (112, 206) where the message is going to be loaded and loading the empty message buffer (114) with the message (see page 3, lines 14-17 and 20-21 and page 6, lines 1-2 of the application).

The claimed subject matter of independent claim 15 pertains to a method for providing interprocessor communication between first and second processors (102, 202; 104, 204) using a shared memory (112, 206) that includes the transmit memories of the first and second processors (102, 202; 104, 204) (see FIGs. 1 and 2 and page 3, lines 3-9 and page 4, lines 12-15 of the application). The first processor (102, 202) is a master processor that is assigned to manage the shared memory (112, 206) (see page 3, lines 3-9 and page 4, lines 12-15 of the application).

The method includes the steps of – at the first processor (102, 202) – allocating a memory buffer (114) from the shared memory (112, 206) for use in loading a message to be sent to the second processor (104, 204) in response to a specific request from the second processor (104) and loading the message in the memory buffer (114) (see FIG. 3 (steps 302 and 304) and page 5, lines 8-11 of the application). The method also includes the steps of sending a message buffer pointer to the second processor (104, 204) and – at the second processor (104, 204) – using the message buffer pointer to locate the message in the shared memory (112, 206) (see FIG. 3 (steps 306, 308, 310) and page 5, lines 11-16 of the application).

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Whether claims 1-3, 7-11, 15, 18 and 19 are patentable under 35 U.S.C. 103(a) over U.S. Patent No. 6,912,716 to Johanson, et al. (Johanson) in view of U.S. Patent No. 6,131,113 to Ellsworth, et al. (Ellsworth).

Whether claims 4-6, 12-14, 16 and 17 are patentable under 35 U.S.C. 103(a) over Johanson in view of Ellsworth and further in view of U.S. Patent No. 6,823,511 to McKenney, et al. (McKenney).

VII. ARGUMENT

A. The recitations of Johanson and Ellsworth do not render the invention of claims 1-3, 7-11, 15, 18 and 19 unpatentable.

As noted above, independent claim 1 recites the limitation that the shared memory includes the transmit memories of both the first and second processors and that the first processor is a master processor that manages the shared memory. It is inherent, then, that the first processor manages the transmit memories of the first and second processors, as one of skill in the art would appreciate. That is, to qualify as a master processor, a processor must manage the transmit memories of both the first and second processors. Independent claims 11 and 15 contain similar limitations.

Applicants submit that Johanson (nor any other prior art reference) does not disclose such a feature. Johanson calls for a shared memory that is separated into three main portions: (1) a first processor to second processor fixed portion; (2) a second processor to first processor fixed portion; and (3) an unallocated dynamic

portion (see col. 4, lines 8-11). The first processor has write access in an allocation starting at the bottom of the shared memory – or portion (1) above - and filling upwards, while the second processor has write access in an allocation starting at the top of the shared memory – portion (2) above – and filling downward (see col. 5, lines 13-18). Depending on the size of the message, either processor can dynamically allocate messages in the unallocated dynamic portion, with the first processor allocating from lower to higher addresses and the second processor in the opposite direction (see col. 4, lines 28-37). Both processors have read access to all portions of the shared memory (see col. 5, lines 18-21). Johanson goes on to describe one possible implementation in which the first processor has access to the dynamic unallocated portion, while the only way the second processor would be allowed to write to this unallocated portion is by requesting write access from the first processor (see col. 6, lines 44-50).

In view of this description, it is clear that neither the first processor nor the second processor is a master processor that manages the shared memory, i.e., a processor that completely manages the shared memory that includes the transmit memories of the first and second processors. In other words, while both processors may have read access to all the shared memory, neither has write access to both fixed portions of the shared memory. The Examiner has determined that these fixed portions of Johanson are equivalent to transmit memories (see page 10, lines 7-8 of the Non-Final Office Action of April 13, 2007), which means that neither processor of Johanson manages the transmit memories

of both processors of Johanson, which is a claim limitation of the present application.

Even when one processor of Johanson is designated as the controller of the unallocated portion, that processor still does not have write access (i.e., allocation) to the fixed portion (i.e., transmit memory) assigned to the other processor. This scheme results in a forced static division of shared memory between the two processors, a disadvantage that the presently claimed subject matter seeks to avoid (see page 1 of the application).

Each of independent claims 1, 11 and 15 also recites the limitation of the first processor sending a message buffer pointer to the second processor that directs the second processor to the message buffer. Applicants submit that Ellsworth does not describe such a feature.

As explained in column 7, lines 23-49 of Ellsworth, the first processor places in a first data element of the resource queue a pointer to an available section of the shared resource. The first processor will then increment a tail pointer for the resource queue. The first processor will continue to place the pointers to the available section in the resource queue and to increment the tail pointer until it determines – by detecting that the tail pointer is pointing to the queue end - that the resource queue is full.

In addition and as described in column 7, line 50 to column 8, line 31, the second processor, will fetch a pointer to the next available portion of the shared resource. The second processor will then appropriately adjust the head pointer

and transmit a "resource consumed" event message to the mailbox message subsystem associated with the first processor. The resource consumed message notifies the first processor that a portion of the shared resource was allocated and that an additional available shared resource section can be placed in the resource queue. As described in column 8, line 51 to column 9, line 3, in response to the resource consumed message, the first processor stores a pointer to an available section of the shared resource in the resource queue location pointed to by the tail pointer. The first processor will also set the tail pointer in the appropriate position.

The claimed limitation that the first processor sends a message buffer pointer to the second processor that directs the second processor to the message buffer is important because it enables the first processor to actively manage the shared memory, while at the same time it eliminates the requirement that the second processor actively manage the shared memory. Specifically, the first processor can receive a request from the second processor and can reserve or allocate a known portion of the shared memory for the second processor, the location of which – in accordance with the claim language - can be forwarded to the second processor. In direct contrast, the second processor in Ellsworth does not receive such a message from the first processor, and as a result, the second processor of Ellsworth must actively search for available portions of the shared resource. That is, the first processor in Ellsworth merely provides pointers to the resource queue, not to the second processor.

Moreover, because the first processor in Ellsworth does not send message

buffer pointers to the second processor, the first processor is unaware of the actual memory needs of the second processor. In other words, in Ellsworth, the first processor blindly makes portions of the shared resource available to the second processor. This method of providing portions of a shared resource is inefficient management of the shared resource. Even worse, this inefficiency is compounded as the number of processors in the system to which the first processor must provide portions of the shared resource increases. Such is not the case with the presently claimed subject matter in view of the first processor knowing the memory needs of any number of processors and the first processor's ability to signal these multiple processors with address pointers. Accordingly, Applicants submit that independent claims 1, 11 and 15, and those claims that depend from them, are patentable over the Johanson and Ellsworth combination.

B. The recitations of Johanson, Ellsworth and McKenney do not render the invention of claims 4-6, 12-14, 16 and 17 unpatentable.

As explained in Section VII(A) above, the combination of Johanson and Ellsworth does not read on independent claims 1, 11 or 15. As such, dependent claims 4-6, 12-14, 16 and 17 are all patentable over the Johanson, Ellsworth and McKenney combination.

Conclusion

Because every element of the claimed subject matter is not disclosed by the cited prior art references, either individually or in combination with one another, Applicants submit that the claims on appeal are patentable.

For the reasons set forth above, and as is apparent from a review of the above-cited references, the claims on appeal present patentable subject matter such that reversal of the rejection is appropriate.

The Commissioner is hereby authorized to charge any requisite fee, or credit any overpayment, to Motorola, Inc., Deposit Account No. 50-2117

Respectfully submitted,

Date: January 23, 2008

By: /Larry G. Brown/
Larry G. Brown

Please send correspondence to:

Motorola, Inc.
1303 E. Algonquin Road
IL01/3RD
Schaumburg, IL 60196

Customer Number: 24273

Attorney for Applicants
Registration No. 45,834

Tel. No.: (954) 723-6449
Fax No.: (847) 576-3750
E-Mail: docketing.us@motorola.com

VIII. CLAIMS APPENDIX

1. (previously presented) An electronic device, comprising:

a first processor;

a second processor coupled to the first processor;

shared memory coupled to the first and second processors, wherein the shared memory includes the transmit memories of both the first and second processors; and

wherein the first processor is a master processor that manages the shared memory and allocates a message buffer to the second processor based on a specific request from the second processor to send a message to the first processor, and wherein the first processor sends a message buffer pointer to the second processor that directs the second processor to the message buffer.

2. (original) An electronic device as defined in claim 1, wherein the first processor sends the message buffer pointer to the second processor in response to receiving an empty buffer request from the second processor.

3. (original) An electronic device as defined in claim 2, wherein after receiving the message buffer pointer the second processor fills the message buffer with the message.

4. (original) An electronic device as defined in claim 3, wherein after filling up the message buffer with the message, the second processor passes the message buffer pointer to the first processor.

5. (original) An electronic device as defined in claim 4, wherein the first processor reads the message from the message buffer after receiving the message buffer pointer.

6. (original) An electronic device as defined in claim 5, wherein after reading the message, the first processor releases the message buffer.

7. (original) An electronic device as defined in claim 1, wherein a plurality of buffers assigned to the second processor are located in the shared memory.

8. (original) An electronic device as defined in claim 7, wherein the plurality of buffers assigned to the second processor are used by the second processor without having to request them from the first processor.

9. (original) An electronic device as defined in claim 8, wherein when the second processor needs to send a message to the first processor it loads a starting address of the message in one of the plurality of buffers assigned to the second processor.

10. (original) An electronic device as defined in claim 1, wherein the electronic device comprises a radio communication device.

11. (previously presented) A method for providing interprocessor communication between first and second processors using a shared memory that includes the transmit memories of the first and second processors, the first processor being a master processor assigned to manage the shared memory, the method comprising the steps of:

(a) sending a request from the second processor requesting an empty message buffer from the shared memory when the second processor needs to send a message to the first processor;

(b) sending a message buffer pointer from the first processor to the second processor in response to the request sent in step (a);

(c) using the message buffer pointer by the second processor to locate the empty message buffer in the shared memory where the message is going to be loaded; and

(d) loading the empty message buffer with the message.

12. (original) A method as defined in 11, further comprising the step of:

(e) sending the message buffer pointer back to the first processor.

13. (original) A method as defined in claim 12, wherein in response to step (e) the first processor performs the step of:

(f) reading the message.

14. (original) A method as defined in claim 13, further comprising the step of:

(g) releasing the empty message buffer once step (f) has been performed.

15. (previously presented) A method for providing interprocessor communication between first and second processors using a shared memory that includes the transmit memories of the first and second processors, the first processor being a master processor assigned to manage the shared memory, the method comprising the steps of:

at the first processor:

(a) allocating a memory buffer from the shared memory for use in loading a message to be sent to the second processor in response to a specific request from the second processor;

(b) loading the message in the memory buffer;

(c) sending a message buffer pointer to the second processor; and

at the second processor:

(d) using the message buffer pointer to locate the message in the shared memory.

16. (original) A method as defined in claim 15, further comprising the step of:

at the second processor:

(e) reading the message; and

(f) sending the message buffer pointer back to the first processor.

17. (original) A method as defined in claim 16, wherein the first processor upon receiving the message buffer pointer sent in step (f), releases the allocated memory buffer so it can be used for a future message.

18. (original) A method as defined in 15, wherein step (c) is performed by the first processor sending the starting address of the allocated memory buffer to a memory located in the second processor.

19. (original) A method as defined in claim 18, wherein the first processor sends an interrupt to the second processor once it has loaded the starting address of the allocated memory buffer in the memory located in the second processor.

IX. EVIDENCE APPENDIX

None

X. RELATED PROCEEDINGS APPENDIX

None